Logotipo, nombre de la empresa

Descripción generada automáticamente

Centro de Ciencias Básicas.

Redes de Computadoras I.

Grado y Grupo: 5to C

Proyecto final: Packet Sniffer

Docente: Javier Santiago Cortes López.

Integrantes: ID:

José Luis Espinoza Sánchez. 280676

Brandon Ruiz Morales. 250476

Héctor Oswaldo Villegas Pérez. 277064

*Índice: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

Temas: Pagina:

Introducción \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_3

Explicación del Código\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_4

Manual de usuario \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_13

Conclusiones \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_23

Referencias \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_25

*Introducción: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

En este documento presentaremos como se creó un analizador de paquetes (packet sniffer) de red utilizando como referencia la serie de artículos publicados en Medium por el autor Talalio que el profesor Santiago nos proporcionó, a través de las siguientes ligas:

[https://talalio.medium.com/building-a-packet-sniffer-9460f394041](https://talalio.medium.com/building-a-packet-sniffer-9460f394041%20)

[https://talalio.medium.com/building-a-packet-sniffer-part-2-6fb33bd68d53](https://talalio.medium.com/building-a-packet-sniffer-part-2-6fb33bd68d53%20)

<https://talalio.medium.com/building-a-packet-sniffer-part-3-a404e60d91c5>

Después de la implementación descrita en los artículos, hemos trasladado el código base hacia una aplicación gráfica utilizando la biblioteca Qt. Permitiendo una representación visual intuitiva de los paquetes capturados, facilitando la visualización y el análisis de los datos en tiempo real.

El objetivo principal de este proyecto es demostrar la captura, procesamiento y despliegue de paquetes de red de manera eficiente, combinando funcionalidades de bajo nivel con una interfaz gráfica amigable. Además, se busca que el desarrollo sea accesible y fácil de entender para aquellos interesados en aprender sobre la captura y análisis de paquetes.

*Explicación del Código: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

Mainwindow.cpp

*Texto

Descripción generada automáticamente*

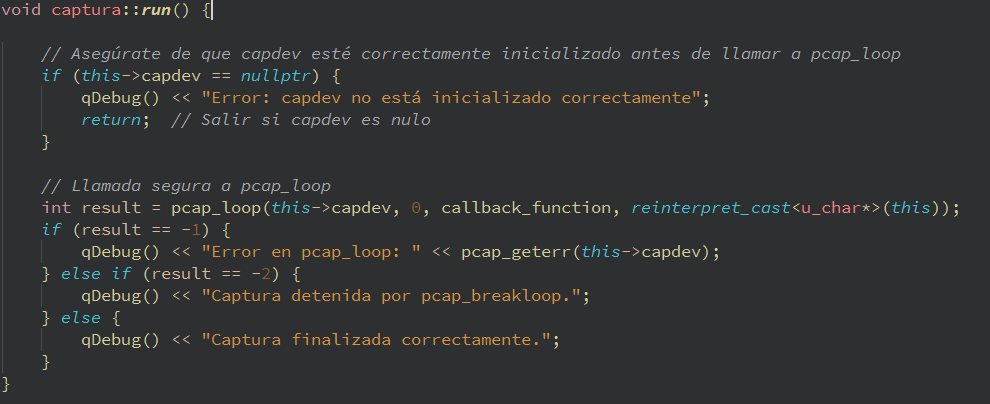
La función MainWindow se utiliza para comparar si se seleccionó una opción diferente de la que está por defecto en el comboBox utilizando un *if* que guarda el nombre de la interfaz en la variable *selectedDevice* y crea un nuevo objeto *packetSniffer()* responsable de la detección de paquetes. En general esta función es para la selección de la interfaz de red, y en caso de no cumplir la condición envía un mensaje de error.

Texto

Descripción generada automáticamente

La función *loadWebInterface()* busca todas las interfaces de red disponibles en el sistema y las carga en un *comboBox* en la ventana de inicio. También agrega una opción predeterminada [ SELECCIONA UNA INTERFAZ DE RED ]. Si ocurre un error al obtener las interfaces, se muestra un mensaje en la consola. Además, la función configura la consola de Windows para usar la codificación UTF-8 para manejar correctamente caracteres especiales.

packetSniffer.cpp



La función *run()* de la clase *captura* se ejecuta dentro de un hilo y tiene como objetivo iniciar la captura de paquetes de red de una interfaz seleccionada. Utiliza la función *pcap\_loop()* para capturar paquetes hasta que se detenga explícitamente o se produzca un error. Primero crea un *if* para validar que el puntero *capdev* sea nulo, si lo es saldrá y en caso contrario comienza la captura de paquetes donde si *result* es igual a 0 la captura será infinita, si *result* es igual a -1 la captura tuvo un error en *pcap\_loop*, si *result* es igual a -2 la captura es detenida por *pcap\_breakloop* y si *result* es igual a cualquier numero diferente a los anteriores la captura será finalizada.

Texto

Descripción generada automáticamente

Esta función se encarga de la creación grafica de la tabla en el espacio *packetsniffer.ui* donde se van a imprimir los datos extraídos de la captura de datos separándolo por *ID Packet, Time, Protocol, Source IP, Destination IP, TTL, Lenght y Extra Info*, y la creación de *rawDataTextEdit* y *rawHexDataTextEdit* que son punteros a los *QTextEdit* definidos en la interfaz que se utilizarán para mostrar los datos crudos y los datos en formato hexadecimal de los paquetes capturados.

Texto

Descripción generada automáticamente

Mientras se escanean paquetes el boton de iniciar y el boton filtros estarán deshabilitados, ademas de limpiar la tabla de paquetes en caso de que esta exista.

Texto

Descripción generada automáticamente

La función *pcap\_findalldevs* de la librería libpcap obtiene todas las interfaces de red disponibles en el sistema y las almacena en una lista enlazada (alldevs).

Si ocurre algún error, se imprime el mensaje de error en consola *Error finding devices*.

Si no ocurre ningun error se imprime la lista de interfaces con su numero de indice.

Texto

Descripción generada automáticamente

Al seleccionar una interfaz de red recorre la lista obtenida y la compara, si el nombre de la interfaz seleccionada coincide entonces se apuntará a dispositivo elegido, y se abre el dispositivo elegido, si ocurre un error se enviará el mensaje *Error opening device* y saldrá

Texto

Descripción generada automáticamente

Es una función de la librería libpcap que devuelve el tipo de enlace del dispositivo de captura (capdev). El tipo de enlace (o datalink type) describe el tipo de encabezado de enlace que se utiliza para transmitir los paquetes en la red.

Texto

Descripción generada automáticamente Texto

Descripción generada automáticamente

Usando libpcap para capturar y analizar el tráfico de red en tiempo real, se crea la función callback\_function que procesa los paquetes capturados, extrayendo información relevante del encabezado IP, como las direcciones IP de origen y destino, el TTL, el protocolo, y otros detalles.

Emite una señal con la información procesada para que la interfaz de usuario, para mostrar los datos de los paquetes en una tabla.

Se valida el tamaño del encabezado IP, se convierte las direcciones IP a cadenas legibles y se manejan posibles errores durante el proceso.

Texto

Descripción generada automáticamente

La funcion *HiloCaptura* gestiona el inicio de la captura de paquetes en un hilo separado, asegurándose de que la UI permanezca activa y actualizada mientras se procesan los paquetes en segundo plano.

Texto

Descripción generada automáticamente

En la función *agregarPaquete()* se crea una condición donde se verifica que el modelo de la tabla esté inicializada correctamente, si no lo está se envía un mensaje a consola.

Texto

Descripción generada automáticamente

Obtenemos la hora actual con precisión en los segundos y nano segundos para mostrar la hora exacta de la captura del paquete y convierte el identificador de paquete (*packetIDStr)* y el TTL (*time to live*) a cadenas de texto *QString.*

Texto

Descripción generada automáticamente

Extrae los primeros 64 bytes del paquete y se muestran en formato hexadecimal en el *rawDataTextEdit* (un *QTextEdit*). Para ver una parte del paquete de forma "cruda", sin interpretación.

En *rawHexDataTextEdit* se agrega información más legible sobre el paquete: ID, direcciones IP de origen y destino, protocolo (TCP/UDP/ICMP) y TTL. Estos datos se muestran en un widget de texto.

Texto

Descripción generada automáticamenteTexto

Descripción generada automáticamente Texto

Descripción generada automáticamente

Se crea un switch procesamiento de los datos según su protocolo (TCP, UDP, ICMP, desconocido). Para TCP y UDP Se extraen los puertos de origen y destino y se formatea esa información como un *QString*, y se llama a la función *insertData()* para agregar la fila a la tabla.

Para los paquetes ICMP, se procesan los valores de tipo y código del encabezado ICMP y finalmente si el protocolo no es TCP, UDP ni ICMP, se marca como "UNKNOWN" y se agrega la fila con esta información.

Texto

Descripción generada automáticamente

Esta función permite al usuario guardar los datos de la tabla en un archivo CSV, incluyendo los encabezados de columna. Utiliza *QFileDialog* para que el usuario elija la ubicación y el nombre del archivo CSV donde se guardarán los datos del archivo si el usuario cancela el diálogo o no selecciona un archivo, la función termina.

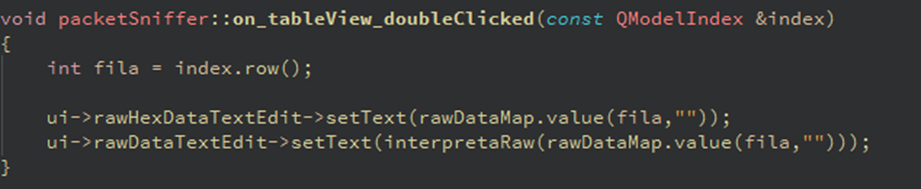
*QTextStream* para escribir los datos de la tabla en formato CSV, recorre todas las columnas de *tableModel* y se obtiene el encabezado de cada columna (usando *headerData* con *Qt::Horizontal* para indicar que se quiere acceder a los encabezados de columna.

Una vez que se han escrito todos los datos, se cierra el archivo se muestra un cuadro de mensaje informando al usuario que el archivo CSV se exportó con éxito.

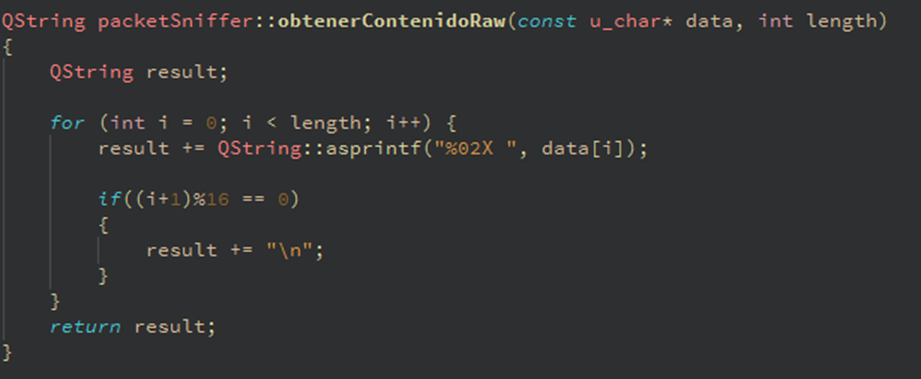
Texto

Descripción generada automáticamente

La función *insertData* agrega una fila con información sobre un paquete a una tabla que se imprimió en la interfaz gráfica. Inserta la información varias columnas según corresponda con información ID del paquete, el tiempo, el protocolo, las direcciones IP de origen y destino, el TTL, la longitud y la información adicional del paquete.



Este método de la clase principalmente cumple con la función de un evento de registrar cuando se hace un doble clic sobre la tabla que muestra los paquetes, tras detectar este evento, muestra en las cajas te texto información relevante sobre el paquete.



En esta función se genera en formato de texto todo el contenido raw del paquete que se está leyendo.

*Manual de usuario: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

1.- Descargue el archivo packet sniffer.exe (ejecutable) y de doble clic para ejecutarlo, al iniciar le aparecerá el siguiente cuadro:

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

2.- Decida si quiere crear un acceso directo en el escritorio dando clic en el cuadro “Crate a desktop shortcut” y de clic en next.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

3.- Se le dará un aviso donde le dirán que todo está listo para instalar, si está de acuerdo de clic en “Install”, si desea cambiar la opción de crear una ventana en el escritorio de clic en back, y si desea cancelar de clic en Cancel.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

4.- Automáticamente se comenzará a instalar, espere hasta que se termine de cargar la barra.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

5.- Si usted quiere arrancar el programa deje el cuadro “Launch Packet Sniffer” marcado y de clic en “Finish”, en caso contrario de clic en cuadro para desmarcarlo y de clic en “Finish”.

Texto

Descripción generada automáticamente

6.- Una vez arrancado el programa en la barra de herramientas se mostrarán las opciones Manual y Ayuda:

6.1- Manual: Al dar clic será Redireccionado a Github, para consultar y descargar este mismo documento.

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente6.2- Ayuda: Al dar clic le abrirá un cuadro donde podrá información sobre los primeros pasos y datos necesarios para la compresión de programa.

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

7.- Seleccione la interfaz de red que desee para comenzar a descargar los paquetes y a continuación de clic en Start Sniffing (nota: Si usted no selecciona ninguna interfaz de red no arrancará el programa).

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

8.- Si usted inicia el programa presionando el boton Start se mostrarán todos los paquetes encontrados (TCP, UDP e ICMP).

8.1 Al lado del boton Start se encuentra el botón Stop, este botón sirve para detener y pausar el flujo de captura de paquetes (NOTA: no hace nada si no está en captura de paquetes).

8.2 Si el flujo de paquetes está detenido y presiona Start de nuevo el flujo de captura de paquetes volverá a iniciar sin necesidad de salir del programa.

Interfaz de usuario gráfica

Descripción generada automáticamente

9.- Después de dar clic en Start (sin filtros), le mostrará una interfaz con tres secciones:

9.1- Tabla: Esta tabla muestra el Id del paquete (ID Packet), la hora precisa de la captura del paquete (time), el tipo de protocolo del paquete (TCP, UDP o ICMP), la dirección Ip de la fuente (Source IP), la Ip Destino (Destination IP), el tiempo de vida del paquete (TTL - Time to Live), el tamaño del paquete (Lenght) y por último Información extra (Extra Info).

9.2- Cuadro inferior izquierdo: Muestra la información de la tabla, pero de manera individual, de cada uno de los paquetes.

9.3- Cuadro inferior derecho: Muestra la información en crudo (sin decodificar), esta información es el flujo de bytes mostrado en hexadecimal.

**Interfaz de usuario gráfica, Sitio web

Descripción generada automáticamente**

10.- La sección de captura muestra 2 distintos apartados que se pueden usar para el beneficio del usuario siendo la posibilidad de usar Filtros dentro del programa para poder seleccionar que elementos son los que queremos consultar dentro de la tabla y para poder crear el CSV con los paquetes capturados.

Pantalla de computadora con números

Descripción generada automáticamente con confianza media

11.- En esta ventana se pueden modificar los filtros a usar en la ejecución mientras se capturan paquetes, en este caso solo se puede elegir un filtro al mismo tiempo. Para elegir activar los filtros se necesita hacer clic sobre el check box de arriba, de lo contrario no habrá algún filtro seleccionado. Tenemos 4 tipos de filtros, para el protocolo, las IP de Origen, IP de Destino y Puerto. Además de poder negar el filtro para hacer capturas con mayor flexibilidad y especificidad.

**Interfaz de usuario gráfica, Texto

Descripción generada automáticamente**

12.- Al hacer dos veces clic sobre un paquete deseado se muestra información relevante del paquete. Principalmente, las direcciones MAC de los dispositivos involucrados, además de en el otro fragmento mostrar el paquete de forma cruda, es decir el flujo de bytes que conforman el paquete, pero ahora mostrado en formato hexadecimal.

Captura de pantalla de computadora

Descripción generada automáticamente

13.-Cuando se da click en el botón de exportar captura, se abre el gestor de archivos donde tendrás que seleccionar un nombre para el archivo que quieras crear de los paquetes capturados.

Tabla

Descripción generada automáticamente

14.- Una vez guardado el archivo CSV se comparte la misma estructura que se puede apreciar en la tabla del programa solo que en este caso lo almacena por distintas celdas y tiene un formato de Excel que es el mejor formato para este caso de capturas ya que así guardamos los datos que necesitemos de forma estructura y fácil de acceder para el usuario.

*Conclusiones: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

Conclusión Jose Luis:

El desarrollo de este proyecto ha sido una buena experiencia de aprendizaje, ya que nos ha permitido comprender cómo interactúan las herramientas de red a bajo nivel con las interfaces de usuario. El hecho de haber migrado el proyecto a una interfaz gráfica utilizando Qt fue fundamental, ya que no solo mejoró la accesibilidad, sino que también optimizó la forma en que el usuario interactúa con los datos. La implementación de una interfaz visual permite que el usuario que no domina estos temas pueda entender y analizar fácilmente los paquetes capturados. Esta mejora no solo hace que la herramienta sea accesible para novatos en redes, sino que también ofrece un punto de partida útil para quienes ya tienen experiencia, permitiendo visualizar y comprender los detalles de las transmisiones en tiempo real de manera más eficiente. Este enfoque es adecuado tanto para entornos académicos, donde los estudiantes pueden aprender sobre redes de manera práctica, como para profesionales de la ciberseguridad, la administración de redes o el análisis forense, quienes pueden beneficiarse de la visibilidad mejorada y la facilidad de uso.

Conclusión Brandon Ruiz:

La implementación de este proyecto fue de un gran agrado porque precisamente el programa a la hora de manejarse por una interfaz gráfica es cuando presenta mayores complicaciones a la hora de que precisamente se buscaba asemejar a una función similar a la que tiene wireshark, el proyecto demuestra precisamente algunos de los conocimientos previos que ya se tenían en C y se tuvieron que desarrollar nuevos por parte de teoría de objetos para trabajar con Qt ya que fue el entorno grafico que mas se adecuo para la creación de este proyecto y realmente fue muy entretenido y gratificante el poder realizarlo ya que entre mas analizas su funcionamiento mas te das cuenta de que el proyecto realmente es mas complejo de lo que parece y que encima se puede hacer todavía mas complejo pero que por falta de tiempo se queda como un proyecto sencillo pero que cumple con su cometido.

Conclusión Héctor Oswaldo:

Este proyecto ha logrado combinar funciones avanzadas de captura y análisis de paquetes con una interfaz fácil de usar, haciendo que la herramienta sea accesible y útil. Al procesar los paquetes en tiempo real, permite a los usuarios detectar problemas o irregularidades en la red de inmediato. Aunque efectivo, el proyecto tiene mucho potencial de mejora. Se podrían añadir filtros avanzados, nuevas opciones de visualización como gráficos o alertas, y optimizar el rendimiento para redes de alto tráfico. También sería interesante ampliar la herramienta a más protocolos y funciones avanzadas, como el análisis de seguridad, convirtiéndola en una solución más completa para el monitoreo de redes

Conclusión general

Este proyecto final no solo nos permitió aplicar conocimientos adquiridos en redes y programación de bajo nivel, sino que también nos desafió a integrar esos conocimientos con tecnologías de interfaz gráfica modernas, como Qt. Nos enfrentamos a retos técnicos tanto en la captura de paquetes como en la visualización de datos, pero el proceso resultó ser enriquecedor y nos brindó una mejor comprensión del flujo de datos en las redes. A lo largo del desarrollo, fue evidente la importancia de ofrecer herramientas poderosas pero accesibles, que simplifiquen tareas complejas y permitan a los usuarios de diferentes niveles de experiencia obtener información valiosa. A futuro, se pueden implementar diversas mejoras para optimizar el rendimiento y expandir la funcionalidad, lo que asegura que esta herramienta pueda seguir siendo útil en una amplia gama de escenarios de análisis de redes

*Referencias\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

<https://www.tcpdump.org/>

<https://npcap.com/>

<https://www.wireshark.org/>

<https://npcap.com/guide/npcap-tutorial.html>

<https://npcap.com/guide/npcap-devguide.html#npcap-examples>

<https://github.com/nmap/npcap/tree/master/Examples>

<https://www.opensourceforu.com/2011/02/capturing-packets-c-program-libpcap/>

<https://www.winpcap.org/docs/docs_411/html/group__wpcap__tut.html>